PRIFYSGOL ABERYSTWYTH UNIVERSITY

Aberystwyth University

Self-adjusting harmony search-based feature selection

Zheng, Ling; Diao, Ren; Shen, Qiang

Published in:
Soft Computing

DOI:
10.1007/s00500-014-1307-8

Publication date:
2015

Citation for published version (APA):
Zheng, L., Diao, R., & Shen, Q. (2015). Self-adjusting harmony search-based feature selection. *Soft Computing*, *19*(6), 1567-1579. https://doi.org/10.1007/s00500-014-1307-8

General rights
Copyright and moral rights for the publications made accessible in the Aberystwyth Research Portal (the Institutional Repository) are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the Aberystwyth Research Portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the Aberystwyth Research Portal

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

tel: +44 1970 62 2400
email: is@aber.ac.uk

Download date: 05. May. 2023

Noname manuscript No.
(will be inserted by the editor)

Self-Adjusting Harmony Search-based Feature Selection

Ling Zheng · Ren Diao · Qiang Shen

Received: date / Accepted: date

Abstract Many strategies have been exploited for the task of feature selection, in an effort to identify more compact and better quality feature subsets. The development of nature-inspired stochastic search techniques allows multiple good quality feature subsets to be discovered without resorting to exhaustive search. In particular, harmony search is a recently developed technique mimicking musicians' experience, which has been effectively utilised to cope with feature selection problems. In this paper, a self-adjusting approach is proposed for feature selection with an aim to further enhance the performance of the existing harmony search-based method. This novel approach includes three dynamic strategies: restricted feature domain, harmony memory consolidation, and pitch adjustment. Systematic experimental evaluations using high dimensional, real-valued benchmark data sets are conducted in order to verify the efficacy of the proposed work.

Keywords Feature Selection · Harmony Search · Harmony Memory Consolidation · Pitch Adjustment Strategy

1 Introduction

Feature selection (FS) [18] is a concept to maximise or preserve a predictive outcome while minimising the dimensionality of given input features [26]. In real-world applications, data often carries many features, which do not necessarily contain any useful information for the problem at hand. When analysing examples in a high-dimensional data problem (e.g., genomic micro-array) [36,44], the performance of a learned predictor becomes weak due to the high inter-dependency amongst individual features, or of combined subsets of features. This is the so-called "curse-of-dimensionality" problem [3]. Applications such as text processing, data classification, and systems control [27,37,38] can benefit from the application of FS, after the noisy, irrelevant, redundant or misleading features being removed or reduced [21].

Given a data set with n dimensions, FS attempts to search for an "optimal" feature subset amongst 2^n candidate subsets. An exhaustive method may be employed. However, it is often impractical for most data sets. Alternative approaches based on hill-climbing (HC) have been exploited where features are removed or added iteratively until there is no further improvement to the current candidate solution. These techniques may converge quickly, but their work on an individual feature basis may lead to a strong assumption that features are entirely independent of each other. Correlated features may potentially lose their required attention, whilst such inter-feature dependences are usually very common in real-world data. Thereby, HC approaches may result in the discovery of sub-optimal solution, both in terms of the quality evaluation score and the subset size. Alternative methods use random search or heuristic strategies in an attempt to overcome such shortcomings. Nature-inspired heuristics such as genetic algorithms (GAs) [17], genetic programming [33], particle swarm optimisation (PSO) [42], simulated annealing [1], and tabu search [31] are also employed for FS with varying degrees of success.

Harmony search (HS) [16] is a recently developed meta-heuristic optimisation algorithm mimicking musical improvisation phenomenon, during which, each musician plays a note for discovering a best harmony all together. HS has been very successful in performing various engineering optimisation problems [2,7,14,25,39,41,46] and machine learning tasks [8,28]. Several advantages over traditional optimisation techniques have also been demonstrated [16,45]. HS imposes only limited mathematical requirements and is not sensitive

L. Zheng · R. Diao · Q. Shen
Department of Computer Science,
Institute of Mathematics, Physics and Computer Science,
Aberystwyth University, UK
E-mail: {liz5, rrd09, qqs}@aber.ac.uk

to the initial value settings. Being a population-based approach, HS works by generating a new vector that encodes a candidate solution, after considering the quality of existing tentative solutions. This is in contrast to the classical genetic algorithms that typically consider only two (parent) vectors in order to produce a new (child) vector. The original HS technique has been improved by methods that dynamically adjust its parameters [16,29], making the algorithm more adaptive to the variance in variable value ranges. Work has also been carried out to analyse the evolution of the population variance over successive generations in HS, thereby drawing important conclusions regarding its exploration power [7].

An application of HS to FS (HSFS) has also been recently developed [12], which has demonstrated competitive FS outcomes. However, the original HSFS is inflexible at adjusting the size of the parameter musician population, which directly affects the performance on feature subset size reduction. This weakness is alleviated to a certain extent by its iterative refinement extension, but the fundamental issue remains. Stochastic mechanisms have not been explored to their maximum potential by the original work, as it does not employ the parameter of pitch adjustment rate due to its ineffective mapping of concepts.

In this paper, a self-adjusting HSFS method is proposed. This new technique extends the original idea of HSFS, built on the basis of an earlier, preliminary investigation into the self-configuration of its internal components [47], supported with substantial rigorous experimental evaluation. In particular, the concept of a restricted feature domain is introduced in order to limit the locally explorable solution domains (of individual musicians), allowing more informative features to be located more quickly, whilst also reducing the run-time memory requirement of the algorithm. A harmony memory consolidation mechanism is developed, which allows musicians (that act as individual feature selectors in the algorithm) to exchange information on tentatively selected features locally, and helps identify and remove non-contributing musicians. As a result, the size of the musician group can be dynamically adjusted during the search. Furthermore, the pitch adjustment strategy is presented which mimics the pitch adjustment behaviour of instrumentalists. It is used by HSFS for fine tuning the emerging feature subsets. In such a scheme, a feature may be substituted by its neighbour, which is determined via the use of a certain feature similarity measure.

The remainder of this paper is organised as follows. Section 2 introduces the original HSFS algorithm that this work aims to improve upon. The proposed self-adjusting HSFS algorithm is described in Section 3, which includes the three newly developed techniques: restricted feature domain, harmony memory consolidation, and pitch adjustment. Section 4 provides results of a series of experimentations, which are carefully conducted for the purpose of demonstrating the benefits and characteristics of the presented strategies. These experimentations include comparative studies with alternative feature selection methods, tested over both typical benchmark datasets and real-world high-dimensional datasets. Finally, Section 5 summarises the paper and suggests the possible directions where the work may be refined and applied.

2 Feature Selection with Harmony Search

HS [16] is a meta-heuristic algorithm that attempts to find a solution vector that optimises a given (possibly multivariate) objective function. During such an iterative search process, each musician (decision variable) chooses a note (variable value) in order to find a best harmony (a potential global optimum) in conjunction with the other musicians. HS has a novel stochastic derivative (for discrete variables) based on musician's experience, rather than gradient (for continuous variables) in differential calculus. In this section, a general overview of the FS concepts is given, and the HS-based FS technique (HSFS) [12] is also summarised for completeness.

An information system in the context of FS is a tuple $\langle X, Y \rangle$, where X is a non-empty set of finite objects, also referred to as the universe of discourse; and Y is a non-empty, finite set of features. For decision systems, $Y = \{A \cup Z\}$ where $A = \{a_1, \cdots, a_{|A|}\}$ is the set of input features, and $|A|$ denotes the cardinality of A, which may be either discrete- or real-valued; and Z is the set of decision features. Various methods have been developed in the literature for the purpose of evaluating the quality of a given feature subset $B \subseteq A$. The focus of this paper lies with the use of group-based feature subset evaluation [9,19,22], which may particularly benefit from the stochastic properties of HSFS [12]. Such evaluation methods typically produce a normalised score $f(B) \in [0, 1]$, $f(\emptyset) = 0$, for a given feature subset B, which is hereafter referred to as the feature subset evaluation score. Here $f : \mathbb{B} \to \mathbb{R}$ represents a subset evaluation function that maps a set of feature subsets onto the set of real numbers.

2.1 Mapping of Key Notions

For conventional optimisation problems, the number of variables is pre-determined by the objective function to be optimised. However for HS, there is not a fixed number of elements in any potential candidate feature subset. In fact, the size of the emerging subset itself should be reduced in parallel to the optimisation of the subset quality evaluation score [9,19,40]. Therefore, when converting concepts, such as those shown in Table 3.2, a musician is best described as an independent expert or a "single feature selector", where the available features translate to musical notes for musicians. Each musician may select one feature to be included in the

emerging feature subset (the harmony), which is the combined vote from all musicians, indicating which features are being nominated.

Table 1 Concept Mapping from HS to FS

HS	Optimisation	FS
Musician	Variable	Feature Selector
Musician Note	Variable Value	Feature
Harmony	Solution Vector	Subset
Harmony Memory	Solution Storage	Subset Storage
Harmony Evaluation	Fitness Function	Subset Evaluation
Optimal Harmony	Optimal Solution	Optimal Subset

The pool of the original features A forms the range of musical notes available to each of the musicians. Multiple musicians are allowed to choose the same feature, and they may opt to choose none at all. A feature subset quality evaluation method assesses each of the new subsets found during the search process. The original HSFS algorithm uses 4 parameters: HMS $|\mathbb{H}|$, the maximum number of iterations $k_{\max}$, the size of the musician group $|M|$, and the *harmony memory considering rate* δ that encourages a musician to randomly choose from all available features (instead of within its own note domain).

Table 2 depicts the following three example harmonies. H^1 denotes a subset of six distinctive features: $B_{H^1} = \{a_1, a_2, a_3, a_4, a_7, a_{10}\}$. H^2 shows a duplication of choices from the first three musicians, and a discarded note (represented by a_-) from musician m_6, representing a reduced subset $B_{H^2} = \{a_2, a_3, a_{13}\}$. H^3 signifies the feature subset $B_{H^3} = \{a_2, a_6, a_4, a_{13}\}$, where $a_3 \rightarrow a_6$ indicates that m_4 originally voted for a_3, but was forced to change the choice to a_6 due to δ activation. For simplicity, the explicit encoding/decoding process between a given harmony H^j and its associated feature subset B_{H^j} is omitted in the following explanation.

Table 2 Feature Subsets Encoding Scheme

	m_1	m_2	m_3	m_4	m_5	m_6	Represented Subset B
H^1	a_2	a_1	a_3	a_4	a_7	a_{10}	$\{a_1, a_2, a_3, a_4, a_7, a_{10}\}$
H^2	a_2	a_2	a_2	a_3	a_{13}	a_-	$\{a_2, a_3, a_{13}\}$
H^3	a_2	a_-	$a_3 \rightarrow a_6$	a_2	a_{13}	a_4	$\{a_2, a_4, a_6, a_{13}\}$

For conventional optimisation problems, the range of possible note choices for each musician is in general different from those for the other musicians. However, when applied to FS, all musicians jointly share one single value range, which is the set of all original features.

2.2 Work Flow of HSFS

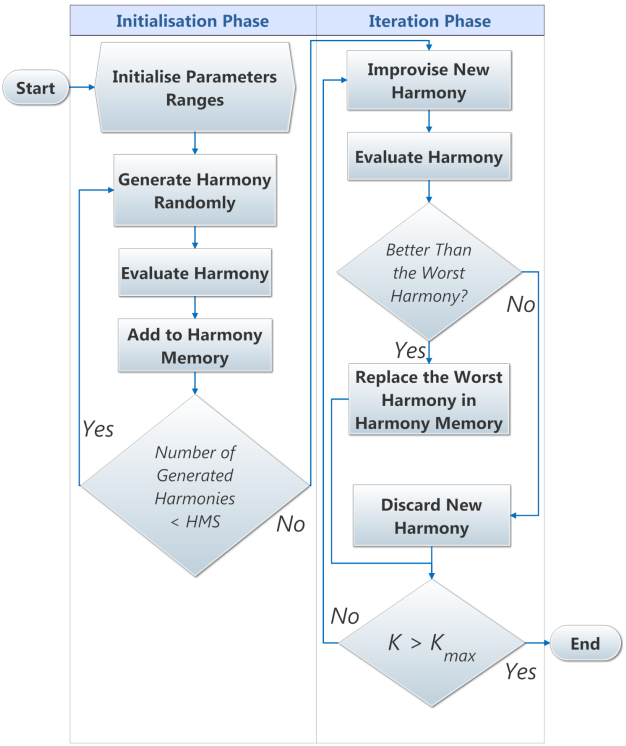

Fig. 1 Proposed Self-Adjusting HSFS Algorithm

The iterative steps of the HSFS algorithm are shown in Fig. 1, where the operations of the approach are outlined.

- Initialise Harmony Memory
  The parameters are assigned according to the given problem, including: $|\mathbb{H}|$, $|M|$, $k_{\max}$, and δ. The subset storage containing $|\mathbb{H}|$ randomly generated subsets is then initialised. This provides each feature selector with a note domain of $|\mathbb{H}|$ features, which may include identical choices and nulls.
- Improvise New Subset
  A new feature is chosen randomly by each feature selector out of their working feature domain, and such chosen features together form a new feature subset. In the event of δ activation, a random feature will be chosen from all available features to substitute the feature selectors' own choice.
- Update Subset Storage
  If the newly obtained subset achieves a higher evaluation score than that of the worst subset in the subset storage, the new subset is recorded and included in the subset storage and the existing worst subset is removed, otherwise this feature subset is discarded. The comparison of subsets takes into consideration of both evaluation score

and subset size in order to discover the minimal reduction at termination.

– Iteration
The improvisation and comparative update procedure continues until a predefined maximum number of iterations $k_{\max}$ is reached. The final output is the feature subset with the highest quality, out of those stored within the harmony memory at termination.

HSFS offers a clear advantage in that a group of features are evaluated as a whole. A newly improvised subset is not necessarily included in the subset storage, simply because one of the features has a locally strong evaluation score. This is the key distinction to any of the hill-climbing based approaches. The integer-valued subset representation also provides a greater freedom of diversification, allowing the stochastic mechanisms of HS to be better exploited. This approach enables the stochastic mechanisms offered by HS to be better exploited, allowing HSFS to achieve a higher search performance than its binary-valued counterpart [11].

3 Self-Adjusting HSFS

The original HSFS algorithm, despite its competitive performance, relies on a limited set of basic procedures to improvise and search for good quality feature subsets. However, the algorithm can potentially be modified to better support FS. This section details three new components developed to enhance the performance of HSFS.

3.1 Restricted Feature Domain

In the original HSFS implementation, all musicians $m_i \in M, i = \{1, \ldots, |M|\}$ jointly use a single domain of values, which is the pool of all features A. The total number of features $|A|$ inevitably affects the rate that musicians identify good quality features. The presence of less informative features also reduces the likelihood of locating better features through δ activation. As a result, the algorithm may potentially spend unnecessary iterations improvising poor quality candidate solutions, and such emerging feature subsets will be discarded since they introduce no improvement to the *harmony memory*.

A new concept termed the *restricted feature domain* (RFD) is therefore proposed to remedy the aforementioned shortcoming. This mechanism restricts the value domain $\aleph_i$ for any given musician m_i to a selective subset of A: The RFDs are constructed during the initialisation phase and reconstructed when the number of musicians is adjusted during the iteration phase. Hence, the recombination of RFDs dynamically affects the choice of musicians throughout the search process. Of course, the union of these RFDs should be equivalent to the full set of features: $\bigcup_{i\in\{1,2,\ldots,|M|\}} \aleph_i = A$ in order to ensure that no important features are mistakenly left out. The feature distribution amongst all the musicians should also be random but uniform. The cardinality of $\aleph_i$ is thus devised to be controlled by a restricted ratio λ, $0 < \lambda \leq 1$, such that $|\aleph_i| = \lceil \lambda \cdot |A| \rceil$.

A set of RFDs can be generated through various methods, so long as the desired properties that are described above are satisfied. Additionally, if problem domain-specific information is available (e.g., provided by human experts), RFDs may also be populated or adjusted in favour of better quality features. In this paper, for simplicity, an RFD is empirically generated by randomly removing features until $|\aleph_i| = \lceil \lambda \cdot |A| \rceil, \lambda = 0.8$, while maintaining a full coverage of features across all musicians. The pseudo code of the suggested mechanism is given in Algorithm 1.

Algorithm 1: HSFS with RFD

```
A: full set of features
M: group of musician m_i ∈ M, i = 1, ..., |M|
δ: harmony search considering rate
H_new: new harmony being improvised
S_i = ⋃_{j=1}^{ℍ} H_i^j: note domain of musician m_i
ℵ_i ← A: RFD of musician m_i
for i ← 1 to |M| do
    while |ℵ_i| > ⌈λ · |A|⌉ do
        ℵ_i ← ℵ_i \ random(ℵ_i)
while ⋃_{i∈{1,...,|M|}} ℵ_i ≠ A do
    for i ← 1 to |M| do
        ℵ_i ← ℵ_i ∪ random(A \ ⋃_{i∈{1,...,|M|}} ℵ_i)
        ℵ_i ← ℵ_i \ random(ℵ_i)
H_new ← ∅
for i ← 1 to |M| do
    if random(1) < δ then
        H_new ← H_new ∪ random(S_i)
    else
        H_new ← H_new ∪ random(ℵ_i)
return H_new
```

3.2 Self-Configuration of Musician Size

The main challenge for FS is to effectively reduce the size of candidate feature subsets, while maintaining their original semantics. The iterative refinement procedure employed by HSFS aims to address this issue (to a certain extent), which is made possible by its flexible mapping of musical concepts onto their associated elements in FS. In particular, a musician is not tied to a specific feature, thereby becoming an independent, single-feature-selector. This forms a sharp contrast to many alternative methods that rely on binary-valued feature subset representation.

Algorithm 2: Process of HMC

```
if |M| ≠ max_{j∈{1,···,|ℍ|}} (|B_{H^j}|) then
    |M| = max_{j∈{1,···,|ℍ|}} (|B_{H^j}|)
    for H^j ∈ ℍ do
        if |M| < |H^j| then
            // Consolidation
            while |M| < |H^j| do
                H^j ← H^j \ a_−
        else
            // Expansion
            while |M| > |H^j| do
                H^j ← H^j ∪ {a_−}
```

The iterative refinement procedure first initialises the number of musicians to be the size of the complete set of original features $|M| = |A|$. This ensures that no human assumption is involved in the configuration of $|M|$, and that the feature subset size may be adjusted according to the actual amount of redundancy present in the data. $|M|$ is then iteratively reduced (in so doing, the size of feature subsets to be selected is restricted) until no smaller solution can be found without sacrificing the evaluation score. As such, although effective, this procedure leads to repetitive executions of the entire search process, and the earlier executions may also over-restrict the search process to a sub-optimal solution region.

Table 3 Consolidation of Harmony Memory

Iteration	Harmony Memory						
k	a_1	a_-	a_2	a_-	a_1	a_2	a_3
	a_2	a_4	a_5	a_-	a_-	a_5	a_5
k (HMC)	a_1	a_-	a_2	a_-	a_-	a_-	a_3
	a_2	a_4	a_5	a_-	a_-	a_-	a_7
$k+1$	a_1	a_2	a_-	a_-	a_3		
	a_2	a_4	a_5	a_-	a_7		

The self-adjusting HSFS algorithm proposed herein embeds an alternative procedure to the aforementioned. It attempts to dynamically and naturally adjust $|M|$ throughout a single execution of the search, via a means of identifying and eliminating the potentially non-contributing musicians (with their note domains fully filled by duplicated nominations or discarded votes a_-). This procedure is referred hereafter as *harmony memory consolidation* (HMC). The pseudo-code of HMC is given in Algorithm 2. In particular, to better determine the presence of non-contributing musicians, the following process needs to be performed:

1. The duplicating nominations within each of the harmonies stored in the harmony memory are replaced by a_-.
2. The desirable value of $|M|$ may then be derived using the formula given below:

$$|M| = \max_{j\in\{1,\cdots,|\mathbb{H}|\}} \{a_i | a_i \in H^j, a_i \neq a_-\} + 1. \quad (1)$$

   Alternatively, $|M|$ may be determined by first converting harmonies $H^j \in \mathbb{H}$ into feature subsets B_{H^j}, and then computing:

$$|M| = \max_{j\in\{1,\cdots,|\mathbb{H}|\}} (|B_{H^j}|) + 1. \quad (2)$$

3. The existing harmonies are trimmed by randomly removing $a_- \in H^j$, until $|H^j| = |M|, H^j \in \mathbb{H}$.
4. The normal HSFS improvisation process is then resumed, on the basis of the newly consolidated harmony memory.

An illustrative example is given in Table 3. The initial harmony memory (at iteration k) consists of several duplicate and discarded nominations, which are identified during the HMC process. For instance, feature subset $(a_1, a_-, a_2, a_-, a_1, a_2, a_3)$ may be changed into $(a_1, a_-, a_2, a_-, a_-, a_-, a_3)$ in this given example. The number of musicians is then reduced to $|M| = \max(3, 4) + 1 = 5$, and the respective harmonies are also trimmed. The resultant harmony memory after consolidation is then used for the next iteration $(k + 1)$.

Table 4 Expansion of Harmony Memory

Iteration	Harmony Memory				
k	a_1	a_2	a_3	a_-	
	a_2	a_7	a_4	a_5	
$k+1$	a_1	$\boldsymbol{a_-}$	a_3	a_-	a_2
	$\boldsymbol{a_-}$	a_7	a_4	a_5	a_2

Note that the HMC procedure may also be utilised to facilitate the expansion of the musician group, because the value of $|M|$ is determined with respect to the size of the largest feature subset in $\mathbb{H}$ as shown in Eq. 2. The rationale behind such a mechanism is to allow larger feature subsets to be nominated, if they may lead to a potentially higher (overall) quality solution. Table 4 details an example of harmony memory expansion, where the harmony memory at iteration k contains a candidate feature subset of size $|B| = |M|$. It is probable that there exists more informative feature subsets of size $|B'| > |M|$, and therefore, the size of the musician group is enlarged by inserting a_- at random positions for all $H \in \mathbb{H}$.

3.3 Feature Subset Adjustment Using Feature Similarity Measures

The base HS algorithm involves another parameter, termed the pitch adjustment rate (PAR). It offers a stochastic mechanism that allows a note chosen by a given musician to be

shifted to a neighbouring one. The underlying motivation of this mechanism is that minor adjustments into neighbouring values may help discover better quality solutions, which is generally true for real-valued optimisation problems. PAR, in conjunction with δ, ensures that fine adjustments can be made to an emerging solution, and the solution region may be sufficiently explored.

The original HSFS algorithm does not exploit the benefits offered by PAR. This is because that the values now represent feature indices, each feature and its "neighbours" may not have such general relation, and thus, an adjustment will result in a change into a possibly unrelated feature nearby. However, the absence of PAR hinders the strength of the algorithm in terms of its effectiveness for finding good quality feature subsets. Having recognised this, the pitch adjustment mechanism is re-introduced in this improved HSFS approach together with a method to determine "neighbouring" features. In the context of FS, the use of the PAR parameter ω, $0 \leq \omega \leq 1$, will enable a musician to choose a neighbouring feature. Such a feature bears a similarity with the original similarity within the range calculated on the basis of the formula $(1-\tau) + Rand(2 \times \tau)$, where τ is the fret width that constrains the maximal amount of dissimilarity allowed. Note that $(1-\omega)$ denotes the probability of using the chosen value without further alteration. Also, as suggested in the original HS algorithm the pitch adjustment procedure and δ activation are set to two mutually exclusive events so that further adjustment is only carried out when a feature is selected from within the harmony memory.

To determine the neighbouring features, a number of existing feature similarity measurements may be employed. For example, the non-parameter test-based approaches such as the Walds-Wolfowitch test [23] may use to detect the closeness of probability distributions of the variables. However, such measures are sensitive to both location and dispersion of the distributions [5,30], and hence, may not be suitable to measure the similarity of features in arbitrary data sets. Alternatively, the dependency between the features may be utilised to calculate the degree of feature similarity. In this paper, two existing dependency measures, both linear (*correlation coefficient-based* [32]) and non-linear (*fuzzy-rough set-based* [22]) are used to obtain the degree of similarities amongst any paired features.

3.3.1 Correlation Coefficient-Based Feature Similarity

Correlation coefficient is a common linear method for measuring the degree of similarity between two random variables. Correlation coefficient ρ between two random variables x and y is formulated as:

$$\rho(x,y) = \frac{\sigma_{xy}}{\sigma_x \sigma_y} \tag{3}$$

$$\sigma_{xy} = \frac{1}{n-1}(\sum_{i=1}^{n} x_i y_i - n\bar{x}\bar{y}) \tag{4}$$

where σ_x and σ_y signify the variance of a variable and σ_{xy} the covariance between two variables. If x and y are entirely correlated, then purely linear relationship exists and $\rho(x,y)$ is ± 1. Independence of x and y implies $\rho(x,y) = 0$. Hence, the quantity can be used as a measure of similarity between two features [20] with the following properties:

1. $|\rho(x,y)| \leq 1$.
2. $\rho(x,y) = 0$ if and only if x and y are linearly correlated.
3. $\rho(x,y) = \rho(y,x)$.
4. If $x^* = ax + b$ and $y^* = cy + d$ for certain constants a, b, c, d, then $\rho(x^*, y^*) = \rho(x,y)$, implying that the similarity measure is not affected by rescaling and transformation of variables.

The correlation coefficient may be sufficient to the strength of feature similarity, but it makes a strong assumption on linear and highly dependent relationships between features. Non-linear patterns are neglected. For example, assume there exists a quadratic relationship between the value of x and y ($x = y^2$), and y is evenly distributed in the range of $[-1, 1]$. The resulting correlation coefficient $\rho(x,y) = 0$, indicating that x and y are independent. However, there in fact exists a strong dependency between them. Thus, an alternative approach may be necessary.

3.3.2 Fuzzy Rough Set-Based Feature Similarity

Fuzzy-rough set theory [10,35] is an extension of traditional rough set theory [34], where two sets: the lower and upper approximation are defined using fuzzy notions [13]. Rough set is centred upon crisp information granulation. In the crisp case, elements either belong to the lower approximation with absolute certainty or not at all. In the fuzzy-rough case, elements may have a membership in the range $[0, 1]$, allowing greater flexibility in handling uncertainty. The concepts of fuzzy lower and upper approximation are defined as follows:

$$\mu_{\underline{R_P}} X(x) = \inf_{y \in \mathbb{U}} I(\mu_{R_P}(x,y), \mu_X(y)) \tag{5}$$

$$\mu_{\overline{R_P}} X(x) = \sup_{y \in \mathbb{U}} T(\mu_{R_P}(x,y), \mu_X(y)) \tag{6}$$

where I is a fuzzy implicator and T a t-norm. R_P is the fuzzy similarity relation induced by the subset of features P:

$$\mu_{R_P}(x,y) = \bigcap_{a \in P} \mu_{R_a}(x,y) \tag{7}$$

with $\mu_{R_a}(x,y)$ being the weight of similarity between instance x and y for feature a. The following three common fuzzy similarity models can be constructed for this purpose.

$$\mu_{R_a}(x,y) = 1 - \frac{|a(x) - a(y)|}{|a_{\max} - a_{\min}|} \tag{8}$$

$$\mu_{R_a}(x,y) = \exp\left(-\frac{(a(x)-a(y))^2}{2\sigma_a^2}\right) \quad (9)$$

$$\mu_{R_a}(x,y) = \max\left(\min\left(\frac{(a(y)-a(x)+\sigma_a)}{\sigma_a}, \frac{(a(x)-a(y)+\sigma_a)}{\sigma_a}\right), 0\right) \quad (10)$$

where σ_a is the standard deviation of feature a, $a_{\max}$ is the maximum value under the feature a, whereas $a_{\min}$ is the minimum value. The fuzzy positive region is defined by:

$$\mu_{POS_{R_P}(Q)}(x) = \sup_{X \in \mathbb{U}/Q} \mu_{\underline{R_P}} X(x) \quad (11)$$

The resulting degree that a set of features Q depends on another set of features P is denoted as $P \Rightarrow Q$ and is defined as:

$$\gamma_P(Q) = \frac{\sum_{x \in \mathbb{U}} \mu_{POS_{R_P}(Q)}(x)}{|\mathbb{U}|} \quad (12)$$

Since this paper only concerns similarity measurements between two arbitrary single features, say, a_i and $a_j \in A$. Eq. 12 may be re-written as:

$$\gamma(a_i, a_j) = \frac{\sum_{x \in \mathbb{U}} \mu_{POS_{R_{\{a_i\}}}(\{a_j\})}(x)}{|\mathbb{U}|} \quad (13)$$

representing the dependency degree of feature a_i upon a_j. This formula is used to locate the suitable closest and most dependent neighbouring feature. To ease the understanding, Algorithm 3 shows the pseudo-code of the improvisation process using pitch adjustment.

Algorithm 3: Improvisation Process using PAR

H_{new}: target harmony being improvised
$S_i = \bigcup_{j=1}^{\mathbb{H}} H_i^j$: note domain of musician m_i
$\aleph_i$: RFD of musician m_i
$\gamma(a_p, a_q)$, feature similarity measure
for $i \leftarrow 1$ **to** $|M|$ **do**
  if *random[0,1]* $< \delta$ **then**
    $a_p =$ random(S_k)
    if *random[0,1]* $< \omega$ **then**
      $\Delta \leftarrow 1 - \tau +$ random(2τ)
      $a_q = \underset{a_q \in A, a_q \neq a_p}{\operatorname{argmin}} (\gamma(a_p, a_q) - \Delta)$
      $H_{new} \leftarrow H_{new} \cup \{a_q\}$
    else
      $H_{new} \leftarrow H_{new} \cup \{a_p\}$
  else
    $H_{new} \leftarrow H_{new} \cup \{$random$(\aleph_i)\}$
return H_{new}

3.4 Self-Adjusting HSFS

The procedure of the complete self-adjusting HSFS algorithm is illustrated in Fig. 2. It incorporates three new modules: RFD construction, harmony memory consolidation, and PAR-based feature subset improvisation. The complexity of the HSFS algorithm is analysed below. The initialisation requires $O(|M| \times |\mathbb{H}|)$ operations to randomly populate the subset storage, and the improvisation process is of the order $O(|M| \times k_{\max})$ because every feature selector needs to produce a new feature at every iteration. Here, $|\mathbb{H}|$ is the subset storage size, $|M|$ is the number of feature selectors, and $k_{\max}$ is the maximum number of iterations. The construction of restricted feature domains requires $O(|M| \times k_{\max} \times \lceil \lambda \times |A| \rceil)$, which occurs in both initialisation phase and iteration phase when the number of musicians is changed. The HMC procedure has a computational complexity of $O(|\mathbb{H}| \times |M|)$ since all stored harmonies need to be examined and consolidated. The additional overhead introduced by the PAR process is largely due to the feature similarity matrix construction, which incurs a cost of up to $O(|A|^2 \times |X|^2)$, where $|X|$ is the total number of training the instances in data set. Thus, the overall algorithm complexity (including the initial cost of computing the feature similarity values) is:

$$O(|A|^2 \times |X|^2) + O(|M| \times (|\mathbb{H}| + \lceil \lambda |A| \rceil) \times k_{max}) \quad (14)$$

4 Experimentation and Discussion

In this section, the results of experimental investigations are reported to demonstrate the capabilities and characteristics of the proposed improvements. Notationally, in the results, HSFS_{SA} stands for the whole algorithm that incorporates all of the newly proposed mechanisms. Section 4.1 presents a comparison against other search methods, including the original HSFS (HSFS_{O}) and two nature-inspired optimisation techniques: genetic algorithms (GAs) [24] and particle swarm optimisation [6]. Two filter-based feature subset evaluators with score values $f(B)$, $0 \leq f(B) \leq 1$: the correlation-based (CFS), and the probabilistic consistency-based (PCFS) are employed. The experimentation uses a total of 10 real-valued UCI benchmark data sets [15], several of which are of high dimensionality and/or contain a large number of objects, thereby presenting reasonably realistic challenges for the proposed techniques. A summary of these data sets is provided in Table 5. Parameter setting in the investigated algorithms are presented in Table 6. Finally, the C4.5 algorithm [43] is adopted due to its popularity to verify the quality of the selected feature subsets from an end classifier learner's perspective, where stratified ten-fold cross-validation is exploited for accuracy validation.

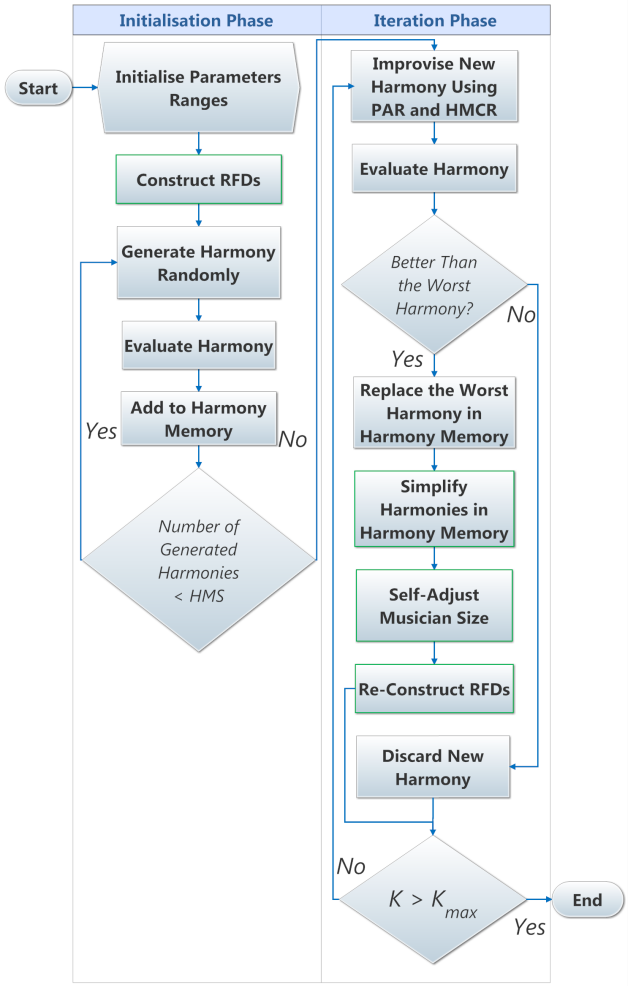

Fig. 2 Proposed Self-Adjusting HSFS Algorithm

Table 5 Data Set Properties

Data Set	Features	Objects	Decisions
arrhythmia	280	452	13
handwritten	257	1593	10
ionosphere	35	230	2
libras	91	360	15
multifeat	650	2000	10
secom	591	1567	2
segment	20	1500	7
sonar	61	208	2
cnae	857	1080	9
web	2557	149	5

4.1 Comparison with Alternative Search Strategies

Stratified tenfold cross-validation (10-FCV) is used. For a given data set, it works by dividing the data into ten sub-tables. Nine of these ten sub-folds are employed for training, where FS is employed to learn the optimal feature subset. The remaining single fold is used to test the classifier using the selected feature subset. This process is then repeated ten times. Therefore, each fold is used for testing only once and the stratification of the data ensures that each class label has the same representation in all folds, thereby helping to alleviate bias/variance problems [4]. In the experiments, 10-FCV is performed using ten different random folds of the data in order to lessen the impact of random factors within the heuristic algorithms. These 10×10 sets of evaluations are aggregated to produce the final experimental outcomes.

Table 6 Parameter Configurations

$HSFS_{SA}$	$\|\mathbb{H}\|$	15
	λ	0.8
	δ	0.8
	k_{max}	5000
$HSFS_{O}$	$\|\mathbb{H}\|$	15
	$\|M\|$	$\|A\|$
	δ	0.8
	k_{max}	5000
GA	Cross Over	0.6
	Populations	20
	Mutation	0.033
	Max Generation	5000
PSO	Particles	100
	C1	1
	C2	2
	Max Generation	5000

In addition, a paired t-test with two-tailed $p = 0.01$ has been performed in order to compare the statistical differences between results obtained by $HSFS_O$ and $HSFS_{SA}$, as given in Table 7. The symbol 'v' indicates that $HSFS_{SA}$ obtains a better results than $HSFS_O$, '$-$' denotes that there exists no statistical difference between the results, and '$*$' signifies that $HSFS_{SA}$ results in statistically worse search performance. These comparisons are made in terms of whether the generated feature subsets offer higher evaluation score, smaller subset size and/or better classification accuracy.

As reflected in Table 7, for CFS, higher evaluation scores are obtained using the proposed techniques for six of the ten data sets (indicated with v) when compared to $HSFS_O$. For the remaining four data sets: *ionosphere*, *secom*, *segment*, and *sonar*, more compact subsets are discovered with equal evaluation scores (or with the tiny difference in evaluation score). The bold figures signify improved performance when compared with HC, where $HSFS_{SA}$ enables further reduction of the size of the selected feature subsets. Although the GAs are capable of identifying higher quality subsets for several data sets, they are unable to identify good quality solutions for the higher dimensional data sets such as *arrhythmia*, *multifeat*, *secom*, *cnae*, and *web*. Importantly, the classification accuracy of the classifier learners built using the reduced feature subsets (selected by $HSFS_{SA}$) is improved, particularly

Table 7 Comparison using CFS and PCFS (10×10) cross-validation, with respect to average subset size, evaluation score, classification accuracy (%), and execution time (millisecond). v, $-$, and $*$ indicate statistically better, equal, and worse results respectively, when compared to the original algorithm. Bold figures signify the overall best results obtained for each of the data sets.

CFS

	Full		$HSFS_{SA}$				$HSFS_O$				GA			PSO			HC	
Dataset	$\|A\|$	C4.5%	Size	Eval.	C4.5%	Time	Size	Eval.	C4.5%	Time	Size	Eval.	Time	Size	Eval.	Time	Size	Eval.
arrhythmia	280	64.38	**12.0**	**0.449** (v)	77.78 (v)	1792	22.9	0.441	62.89	1138	46.2	0.397	5418	13.7	0.288	1371	25	0.438
handwritten	257	75.83	**29.3**	**0.524** (v)	76.39 (v)	1650	63.8	0.510	75.45	954	93.9	**0.523**	8044	127.3	0.472	3416	75	**0.524**
ionosphere	35	87.83	**8.7** (v)	**0.536**	89.56 (v)	46	11.1	0.534	88.43	37	10.3	**0.539**	180	10.3	0.535	195	11	0.521
libras	91	69.72	**15.4**	**0.613** (v)	68.00 ($*$)	817	27.3	0.604	68.04	152	30.5	**0.611**	984	26.5	0.592	435	26	0.608
multifeat	650	94.75	**83.1**	0.920 (v)	95.12 (v)	9413	81.6	0.906	93.50	5587	217.9	0.894	59173	315.2	0.854	27802	144	**0.925**
secom	591	89.60	**15.2** (v)	**0.096**	90.44 (v)	6754	36.8	**0.096**	90.11	4557	51.2	0.012	28847	27.5	0.013	16035	17	**0.096**
segment	20	95.73	**4.5** (v)	**0.734**	94.80 ($-$)	26	5.1	**0.732**	94.80	17	**4.4**	**0.732**	80	**4.4**	0.735	132	6	0.725
sonar	61	71.15	**9.4** (v)	0.352	76.19 (v)	76	17.9	0.352	73.72	80	17.5	**0.359**	412	11.4	0.327	290	19	0.352
cnae	857	88.8	14.4	**0.426** (v)	77.80 (v)	155363	18.6	0.418	69.80	139857	170.9	0.037	65066	**10.8**	0.073	23613	28	0.414
web	2557	51.6	**33.3**	0.543 (v)	57.03 (v)	97512	35.4	0.526	55.33	135671	96.0	0.327	169760	100.9	0.129	27068	59	**0.565**

PCFS

	Base		$HSFS_{SA}$				$HSFS_O$				GA			PSO			HC	
Dataset	$\|A\|$	C4.5%	Size	Eval.	C4.5%	Time	Size	Eval.	C4.5%	Time	Size	Eval.	Time	Size	Eval.	Time	Size	Eval.
arrhythmia	280	64.38	29.1 (v)	**0.989**	67.59 (v)	4202	136.2	**0.989**	64.16	4336	29.2	**0.989**	34957	112.5	**0.989**	4514	**21**	0.988
handwritten	257	75.83	70.2 (v)	**1.000**	79.38 (v)	155	80.0	**1.000**	75.07	140	33.0	**1.000**	127	23.5	**1.000**	5463	**18**	**1.000**
ionosphere	35	87.83	**6.8** (v)	**0.997**	87.10 (v)	444	21.5	**0.997**	86.70	394	10.0	**0.997**	85	8.9	**0.997**	334	**7**	0.996
libras	91	69.72	**16.4**	**0.978** (v)	68.75 (v)	1198	57.2	0.974	69.33	1234	17.3	0.974	11559	28.2	0.973	1099	17	0.969
multifeat	650	94.75	9.1 (v)	**1.000**	95.58 (v)	424	26.2	**1.000**	94.35	407	43.0	**1.000**	222	12.2	**1.000**	10535	**7**	**1.000**
secom	591	89.60	23.7 (v)	**0.990**	89.42 (v)	29084	154.9	**0.989**	88.70	30858	92.2	**0.989**	291034	284.8	**0.990**	30571	**1**	0.936
segment	20	95.73	**6.4** (v)	**0.998**	95.91 (v)	3044	8.5	0.997	95.6	2710	7.0	0.997	2152	7.5	0.997	597	9	**0.998**
sonar	61	71.15	**11.7**	**1.000** (v)	74.72 (v)	552	14.5	0.989	73.2	534	**12.0**	0.989	5132	18.5	0.989	579	14	0.981
cnae	857	88.8	**65.2** (v)	**0.983**	85.19 (v)	122361	69.0	**0.981**	76.85	106798	213.9	**0.983**	324843	699.1	**0.983**	52925	67	**0.981**
web	2557	51.6	**18.6** (v)	**1.000**	62.22 (v)	101459	21.1	**1.000**	54.45	94431	1036.8	**1.000**	778	585.7	**1.000**	26575	21	**1.000**

for *arrhythmia* (by 13.40%), *sonar* (by 5.04%), and *web* (by 5.43%).

Note that the execution time of $HSFS_{SA}$ is longer than $HSFS_O$ for most of the data sets except *sonar* and *web*, which is caused by the complexities incurred because of the introduced self-adjusting components. This observation confirms the complexity analysis performed in Section 3.4.

For the results obtained using the PCFS evaluator, the performance improvement over the original algorithm is more obvious, where better subsets are selected across all data sets. Note that only a single feature is selected by HC for the *secom* data set. This is because no additional features offer any increase in the evaluation score, when combined with this selected feature. However, better combinations of features do exist, which are successfully identified by all other employed stochastic approaches. Feature subsets obtained by GAs are mostly comparable to the others in terms of the evaluation scores. GAs are indeed able to identify feature subsets of optimal quality for data set *libras*. The classifiers built using the feature subsets selected by $HSFS_{SA}$ also show improved classification performance for six of the ten data sets. Interestingly, higher classification accuracies are obtained by $HSFS_{SA}$ for nine of the whole ten data sets when compared to $HSFS_O$. For the remaining data set, *libras*, the minor differences in accuracy are acceptable, given the substantial reduction in the averaged feature subset size (16.40 for $HSFS_{SA}$, and 57.23 for $HSFS_O$).

4.2 Effect of Individual Strategy

Two of the data sets with the largest number of features: *multifeat* and *secom* are employed for the remainder of the experimentation. Each of the proposed modifications to HSFS is tested on its own in order to ascertain what benefits these strategies offer individually. Fig. 3 illustrates the effects of the restricted ratio λ which controls the size of the RFDs. The results are collected using different values for λ ranging from 0.1 to 1, with an interval of 0.1. Intuitively, if the value of λ is too small, the musicians may have too few features to work with, which will limit the solution quality. For both data sets, the quality of the selected subsets approach peaks at $\lambda = 0.8$, when 80% of features in A are utilised.

Fig. 4 aims to demonstrate the effectiveness of the HMC process. The algorithm successfully adjusts the number of musicians to a reasonable level, down from the initial setting of $|M| = |A|$, without subjective intervention. The reduced group size further encourages the remaining feature selectors to identify even smaller candidate solutions. The compactness of the resulting subsets as reported in Tables 7 also provides good evidence for the positive impacts of this process.

The final set of experiments is carried out in order to study the effects of the pitch adjustment rate, by varying ω from 0 to 1 with an interval of 0.1. This parameter manipulates the shift ratio, where a feature is replaced with a random close neighbouring feature. Fig. 5 shows the size

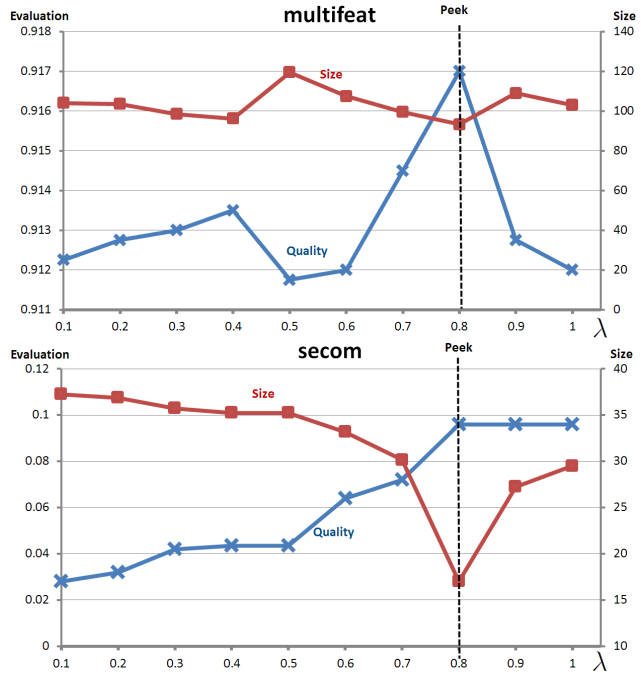

Fig. 3 Demonstration of the Effects of RFD using Different $\lambda \in [0.1, 1]$

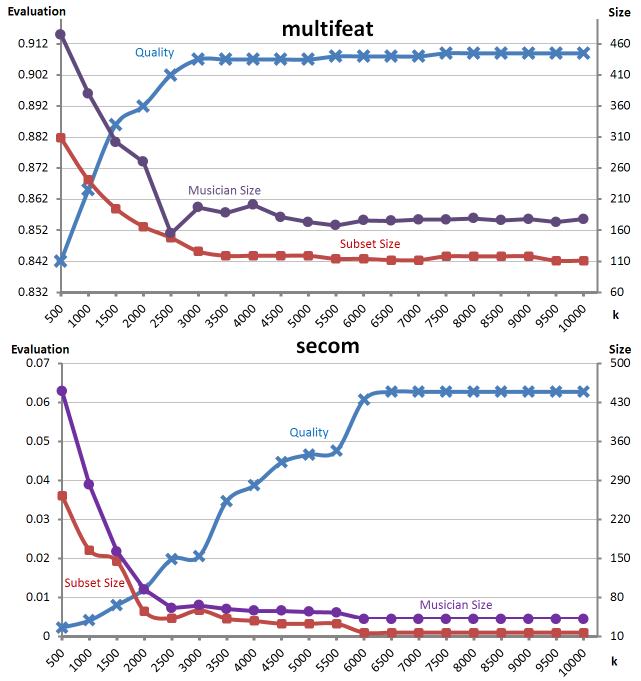

Fig. 4 Automatic Configuration of Musician Size using HMC

and evaluation scores of the best solutions recorded during the search process. The performance of both data sets: *multifeat* and *secom* peaks at $\omega = 0.2$, when a feature shifts to neighbouring features with a probability of 20%. Obviously, if the value of ω is set too large, good quality features may be neglected.

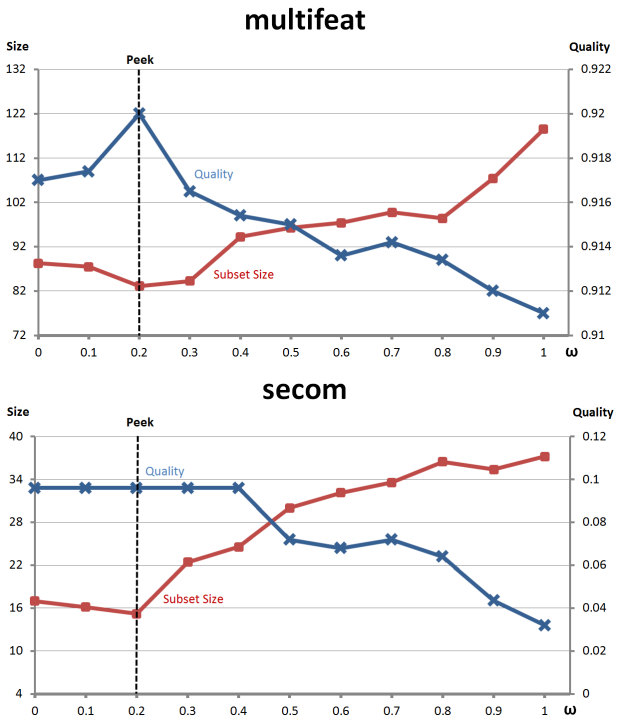

Fig. 5 Demonstration of the Effects of PAR using Different $\omega \in [0, 1]$

5 Conclusion

This paper has presented a self-adjusting FS search algorithm which improves the original harmony search-based feature selection [12] with three new techniques. The proposed techniques are conceptually simple and require limited computational overheads in order to achieve positive effects. The musicians in HSFS_{SA} improvise new candidate feature subsets using a selective portion of the full set of original features (RFD), and dynamically adjust their subset size via the HMC process. The pitch adjustment strategy, re-introduced to HSFS via feature similarity measures allows finer yet relevant adjustments to emerging feature subsets. Experimental results show that the added enhancements can indeed further improve the quality of the resultant feature subsets (when compared to the original HSFS approach), and also obviate the need to precisely pre-configure the size of the musician group.

Although promising, much can be done to further refine the proposed improvements. While the possibility of exchanging information between musicians has been explored in the HMC procedure, there may exist alternative applications of this mechanism (e.g., promoting high quality features, or preserving minority features). Note that these proposed improvements may also be employed by other nature-inspired FS search algorithms. In particular, the PAR mechanism of HS is conceptually similar to mutation operators used by GAs and PSO. Alternative feature similarity measures are

also worth investigating, which may prove to be more efficient than fuzzy-rough set-based measures. Finally, theoretical extensions to the techniques in the area of dynamic FS, classifier ensemble reduction with FS, and feature grouping are also of interest in further developing this work.

Acknowledgements The first author is supported by an Aberystwyth University PhD scholarship, and the second by an Aberystwyth University research fellowship. The authors are grateful to the reviewers who provided the constructive comments on the original version of this paper, which have helped significantly in modifying this research.

References

1. Aarts, E., Laarhoven, P.v.: Simulated annealing: an introduction. Statistica Neerlandica **43**(1), 31–52 (1989)
2. Al-Betar, M., Khader, A., Zaman, M.: University course timetabling using a hybrid harmony search metaheuristic algorithm. IEEE Trans. Syst., Man, Cybern. C **42**(5), 664–681 (2012)
3. Bellman, R.: Dynamic programming, 1 edn. Princeton University Press, Princeton, NJ, USA (1957)
4. Bengio, Y., Grandvalet, Y.: No unbiased estimator of the variance of k-fold cross-validation. Machine Learning Research **5**, 1089–1105 (2004)
5. Bhattacharya, R., Patrangenaru, V.: Nonparametic estimation of location and dispersion on riemannian manifolds. Journal of Statistical Planning and Inference **108**(1), 23–35 (2002)
6. Chuang, L.Y., Chang, H.W., Tu, C.J., Yang, C.H.: Improved binary {PSO} for feature selection using gene expression data. Computational Biology and Chemistry **32**(1), 29 – 38 (2008)
7. Das, S., Mukhopadhyay, A., Roy, A., Abraham, A., Panigrahi, B.: Exploratory power of the harmony search algorithm: analysis and improvements for global numerical optimization. IEEE Trans. Syst., Man, Cybern. B **41**(1), 89–106 (2011)
8. Das Sharma, K., Chatterjee, A., Rakshit, A.: Design of a hybrid stable adaptive fuzzy controller employing lyapunov theory and harmony search algorithm. IEEE Trans. Control Syst. Technol. **18**(6), 1440–1447 (2010)
9. Dash, M., Liu, H.: Consistency-based search in feature selection. Artif. Intell. **151**(1-2), 155–176 (2003)
10. De Cock, M., Cornelis, C., Kerre, E.E.: Fuzzy rough sets: the forgotten step. IEEE Trans. Fuzzy Syst. **15**(1), 121–130 (2007)
11. Diao, R., Shen, Q.: Two new approaches to feature selection with harmony search. In: IEEE Interantional Conference on Fuzzy Systems, pp. 1–7 (2010)
12. Diao, R., Shen, Q.: Feature selection with harmony search. IEEE Trans. Syst., Man, Cybern. B **42**(6), 1509–1523 (2012)
13. Dubois, D., Prade, H.: Putting rough sets and fuzzy sets together. Intelligent Decision Support, Kluwer Academic Publishers, Dordrecht, (1992)
14. Fesanghary, M., Mahdavi, M., Minary-Jolandan, M., Alizadeh, Y.: Hybridizing harmony search algorithm with sequential quadratic programming for engineering optimization problems. Computer Methods in Applied Mechanics and Engineering **197**(33-40), 3080–3091 (2008)
15. Frank, A., Asuncion, A.: UCI machine learning repository (2010)
16. Geem, Z.W. (ed.): Recent advances in harmony search algorithm, *Studies in Computational Intelligence*, vol. 270. Springer (2010)
17. Grefenstette, J.J.: Optimization of control parameters for genetic algorithms. IEEE Trans. Syst., Man, Cybern. **16**(1), 122–128 (1986)
18. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. J. Mach. Learn. Res. **3**, 1157–1182 (2003)
19. Hall, M.A.: Correlation-based feature subset selection for machine learning. Ph.D. thesis, University of Waikato, Hamilton, New Zealand (1998)
20. Hsu, H.H., Hsieh, C.W.: Feature selection via correlation coefficient clustering. Journal of Software **5**(12), 1371–1377 (2010)
21. Jensen, R., Shen, Q.: Are more features better? a response to attributes reduction using fuzzy rough sets. IEEE Trans. Fuzzy Syst. **17**(6), 1456–1458 (2009)
22. Jensen, R., Shen, Q.: New approaches to fuzzy-rough feature selection. IEEE Trans. Fuzzy Syst. **17**(4), 824–838 (2009)
23. Johnson, N.: Linear statistical inference and its applications. Technometrics **8**(3), 551–553 (1966)
24. Leardi, R., Boggia, R., Terrile, M.: Genetic algorithms as a strategy for feature selection. Journal of Chemometrics **6**(5), 267–281 (1992)
25. Lee, K.S., Geem, Z.W.: A new structural optimization method based on the harmony search algorithm. Computers & Structures **82**(9), 781–798 (2004)
26. Liu, H., Motoda, H.: Computational methods of feature selection. Chapman & Hall/CRC (2008)
27. Mac Parthaláin, N., Jensen, R., Shen, Q., Zwiggelaar, R.: Fuzzy-rough approaches for mammographic risk analysis. Intell. Data Anal. **14**(2), 225–244 (2010)
28. Mahdavi, M., Chehreghani, M.H., Abolhassani, H., Forsati, R.: Novel meta-heuristic algorithms for clustering web documents. Applied Mathematics and Computation **201**(1-2), 441–451 (2008)
29. Mahdavi, M., Fesanghary, M., Damangir, E.: An improved harmony search algorithm for solving optimization problems. Applied Mathematics and Computation **188**(2), 1567–1579 (2007)
30. Maronna, R.A., Martin, R.D., Yohai, V.J.: Robust statistics. J. Wiley (2006)
31. Mashinchi, M., Orgun, M.A., Mashinchi, M., Pedrycz, W.: A tabu–harmony search-based approach to fuzzy linear regression. IEEE Trans. Fuzzy Syst. **19**(3), 432–448 (2011)
32. Mitra, P., Murthy, C., Pal, S.K.: Unsupervised feature selection using feature similarity. IEEE Trans. Pattern Anal. Mach. Intell. **24**(3), 301–312 (2002)
33. Muni, D.P., Pal, N.R., Das, J.: Genetic programming for simultaneous feature selection and classifier design. IEEE Trans. Syst., Man, Cybern. B **36**(1), 106–117 (2006)
34. Pawlak, Z.: Rough set approach to knowledge-based decision support. European journal of operational research **99**(1), 48–57 (1997)
35. Radzikowska, A.M., Kerre, E.E.: A comparative study of fuzzy rough sets. Fuzzy Sets and Systems **126**(2), 137–155 (2002)
36. Shah, M., Marchand, M., Corbeil, J.: Feature selection with conjunctions of decision stumps and learning from microarray data. IEEE Trans. Pattern Anal. Mach. Intell. **34**(1), 174–186 (2012)
37. Shang, C., Barnes, D.: Fuzzy-rough feature selection aided support vector machines for mars image classification. Computer Vision and Image Understanding **117**(3), 202–213 (2013)
38. Shen, Q., Jensen, R.: Selecting informative features with fuzzy-rough sets and its application for complex systems monitoring. Pattern Recognition **37**(7), 1351–1363 (2004)
39. Srinivasa Rao, R., Narasimham, S.V.L., Ramalinga Raju, M., Srinivasa Rao, A.: Optimal network reconfiguration of large-scale distribution system using harmony search algorithm. IEEE Trans. Power Syst. **26**(3), 1080–1088 (2011)
40. Tsang, E.C., Chen, D., Yeung, D.S., Wang, X.Z., Lee, J.: Attributes reduction using fuzzy rough sets. IEEE Trans. Fuzzy Syst. **16**(5), 1130–1141 (2008)
41. Vasebi, A., Fesanghary, M., Bathaee, S.M.T.: Combined heat and power economic dispatch by harmony search algorithm. International Journal of Electrical Power & Energy Systems **29**(10), 713–719 (2007)
42. Wang, X., Yang, J., Teng, X., Xia, W., Jensen, R.: Feature selection based on rough sets and particle swarm optimization. Pattern Recognition Letters **28**(4), 459–471 (2007)

43. Witten, I.H., Frank, E.: Data mining: practical machine learning tools and techniques, second edn. Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann (2005)
44. Xing, E.P., Jordan, M.I., Karp, R.M.: Feature selection for high-dimensional genomic microarray data. In: Proceedings of the Eighteenth International Conference on Machine Learning, pp. 601–608. Morgan Kaufmann (2001)
45. Yang, X.S.: Harmony search as a metaheuristic algorithm. In: Music-inspired harmony search algorithm, pp. 1–14. Springer (2009)
46. Zhang, R., Hanzo, L.: Iterative multiuser detection and channel decoding for ds-cdma using harmony search. Signal Processing Letters, IEEE **16**(10), 917–920 (2009)
47. Zheng, L., Diao, R., Shen, Q.: Efficient feature selection using a self-adjusting harmony search algorithm. In: 2013 13th UK Workshop on Computational Intelligence, pp. 167–174 (2013)